Smart Manufacturing Enterprise Rating System General Test Document

1 Introduction

1.1 Document ID

Auth	Date	Version
袁易锋	2021-02-19	v1.0

1.2 Range

This document covers the input and expected results of unit testing, integration testing, and functional testing of the smart manufacturing enterprise rating system.

1.3 Reference

"Software Engineering and Computing (Volume 2) Technical Basis of Software Development"

2 Details

2.1 Summary of total test results

The unit test, integration test, and function test of the rating system for intelligent manufacturing enterprises were completed in turn. The test results were in line with expectations. The product generally fulfilled the requirements of enterprise or individual user registration and login, updating financial indicators, and updating two indicators.

2.2 Detailed test results

2.2.1 Unit test

2.2.1.1 Summary of test results

The project has been unit tested, the test mainly covers the interface of the “dao” layer, and the test ensures that the interface specification and logic of the “dao” layer meet the expectations.

2.2.1.2 Detailed unit test results

Test case ID	Species	Condition	Desired result	Test Result	Test object ID
TUS1-1	Enterprise	EnterpriseMapper.getEnterpriseById(1);	Get the Enterprise object with id number 1 in the database	Get Enterprise object information	GetEnterpriseById
TUS1-2	Enterprise	EnterpriseMapper.getEnterpriseByEmail("leihuang@163.com");	Get the Enterprise object whose email in the database is "leihuang@163.com"	Get Enterprise object information	GetEnterpriseByEmail
TUS1-3	Enterprise	Enterprise enterprise = new Enterprise(); EnterpriseMapper.insertEnterprise(enterprise);	Insert an Enterprise object information into the database, return 1	Get 1	InsertEnterprise
TUS1-4	Enterprise	EnterpriseMapper.deleteEnterprise(1);	Delete an Enterprise object with id 1 from the database, return 1	Get 1	DeleteEnterprise
TUS1-5	Enterprise	Enterprise enterprise = new Enterprise(); EnterpriseMapper.updateEnterprise(enterprise);	Update Enterprise information in the database, return 1	Get 1	UpdateEnterprise
TUS1-6	Enterprise	List Enterprises = EnterpriseMapper.getTwentyEnterprises();	Get up to 20 Enterprise objects in the database and return them in the form of a list	Get a list of Enterprise	GetTwentyEnterprises
TUS1-7	Enterprise	List Enterprises = EnterpriseMapper.getEnterpriseByKey("AI");	Get the Enterprise object with the keyword "AI" in the database, and return it in the form of a list	Get a list of Enterprise	GetEnterpriseByKey
TUS1-8	Enterprise	List list = new List<>(); EnterpriseMapper.updateEpList(list);	Update Enterprise information in batches in the database, return 1	Get 1	UpdateEpList
TUS1-9	Enterprise	FinancialIndex financialIndex = new FinancialIndex（）； FinanIndMapper.insertFinanInd(financialIndex);	Insert financial indicators into the database, return 1	Get 1	InsertFinanInd
TUS1-10	Enterprise	FinanIndMapper.getFinanByEpId(1);	Obtain the financial index information of the financial index of the enterprise id 1 in the database	Get the FinancialIndex object	GetFinanByEpId
TUS1-11	Enterprise	List finInds = FinanIndMapper.getAllFinInd();	Get all the FinancialIndex objects in the database and return them in the form of a list	Get a list of FinancialIndex	GetAllFinInd
TUS1-12	Enterprise	FinanIndMapper.deleteFinanById(1);	Delete the FinancialIndex with id 1 from the database, and return 1	Get 1	DeleteFinanById
TUS1-13	Enterprise	MassDiscrete massDiscrete = new MassDiscrete（）； MassDisMapper.insertMassDis(assDiscrete);	Insert the MassDiscrete indicator MassDiscrete into the database and return 1	Get 1	InsertMassDis
TUS1-14	Enterprise	MassDisMapper.getMassDisByEpId(1);	Obtain the information of the MassDiscrete discrete mass enterprise two-dimensionalization index information with the enterprise id of 1	Get the MassDiscrete object	GetMassDisByEpId
TUS1-15	Enterprise	List assDiscrete = MassDisMapper.getAllMassDis();	Get all MassDiscrete objects in the database and return them as a list	Get a MassDiscrete list	GetAllMassDis
TUS1-16	Enterprise	MassDisMapper.deleteDiverById(1);	Delete MassDiscrete with id 1 from the database, return 1	Get 1	DeleteDiverById

Test case ID	Species	Condition	Desired result	Test Result	Test object ID
TUS1-17	Enterprise	MixIndustry mixIndustry = new MixIndustry（）; MixInduMapper.insertMixIndu(mixIndustry);	Insert the MixIndustry indicator MixIndustry into the database and return 1	Get 1	InsertMixindu
TUS1-18	Enterprise	MixInduMapper.getMixIndByEpId(1);	Obtain the information of the two indicators of the MixIndustry mixed industry enterprise with the enterprise id of 1	Get the MixIndustry object	GetMixIndByEpId
TUS1-19	Enterprise	List mixIndustry = MixInduMapper.getAllMixInd();	Get all the MixIndustry objects in the database and return them in the form of a list	Get a MixIndustry list	GetAllMixInd
TUS1-20	Enterprise	MixInduMapper.deleteDiverById(1);	Delete MixIndustry with id 1 from the database, return 1	Get 1	DeleteDiverById
TUS1-21	Enterprise	ProcIndustry procIndustry = new ProcIndustry（）; ProcInduMapper.insertProcIndu(procIndustry);	Insert ProcIndustry into the database, and return 1	Get 1	InsertProcindu
TUS1-22	Enterprise	ProcInduMapper.getProcByEpId(1);	Obtain the ProcIndustry process industry and industrialization index information with the company id of 1 in the database	Get ProcIndustry object	GetProcByEpId
TUS1-23	Enterprise	List procIndustry = ProcInduMapper.getAllProcInd();	Get all ProcIndustry objects in the database and return them in the form of a list	Get a ProcIndustry list	GetAllProcInd
TUS1-24	Enterprise	ProcInduMapper.deleteDiverById(1);	Delete ProcIndustry with id 1 from the database, return 1	Get 1	DeleteDiverById
TUS1-25	Enterprise	ServeIndustry serveIndustry = new ServeIndustry（）; ServInduMapper.insertServIndu(serveIndustry);	Insert the service industry dualization indicator ServeIndustry into the database, and return 1	Get 1	InsertSerindu
TUS1-26	Enterprise	ServInduMapper.getSerIndByEpId(1);	Obtain the information of the two indicators of the ServeIndustry service industry with the enterprise id of 1 in the database	Get the ServeIndustry object	GetSerIndByEpId
TUS1-27	Enterprise	List serveIndustry = ServInduMapper.getAllSerInd();	Get all ServeIndustry objects in the database and return them in the form of a list	Get a ServeIndustry list	GetAllSerInd
TUS1-28	Enterprise	ServInduMapper.deleteDiverById(1);	Delete ServeIndustry with id 1 from the database, return 1	Get 1	DeleteDiverById
TUS1-29	Enterprise	SmeDiscrete smeDiscrete = new SmeDiscrete（）; SmeDisMapper.insertSmeDis(smeDiscrete);	Insert the SmeDiscrete indicator of the small and medium-sized discretization industry into the database, and return 1	Get 1	InsertSmeDis
TUS1-30	Enterprise	SmeDisMapper.getSmeDisByEpId(1);	Obtain the SmeDiscrete small and medium-sized discretization industry two indicators information in the database with the company id of 1.	Get the SmeDiscrete object	GetSmeDisByEpId

Test case ID	Species	Condition	Desired result	Test Result	Test object ID
TUS1-31	Enterprise	List smeDiscrete= SmeDisMapper.getAllSmeDis();	Get all SmeDiscrete objects in the database and return them in the form of a list	Get a SmeDiscrete list	GetAllSmeDis
TUS1-32	Enterprise	SmeDisMapper.deleteDiverById(1);	Delete the SmeDiscrete with id 1 from the database, and return 1	Get 1	DeleteDiverById
TUS2-1	User	UserMapper.getUserById(1);	Get the User object with id 1 in the database	Get the Uesr object	GetUserById
TUS2-2	User	UserMapper.getUserByEmail("123456@qq.com");	Get the User object whose email is "123456@qq.com" in the database	Get the Uesr object	GetUserByEmail
TUS2-3	User	User user = new User（）； UserMapper.createNewUser(user);	Create a new user and insert it into the database, return 1	Get 1	CreateNewUser
TUS2-4	User	UserMapper.deleteUserById(1);	Delete the User information with id 1 from the database, and return 1	Get 1	DeleteUserById
TUS2-5	User	User user = new User（）； UserMapper.updateUser(user);	Update User information in the database, return 1	Get 1	UpdateUser
TUS2-6	User	UserMapper.getUserNum();	Get the number of users	Get the number of users (int)	GetUserNum
TUS2-7	User	Browse browse = new Browse（）； BrowseMapper.insertBrowse(browse);	Insert a browsing record into the database, return 1	Get 1	InsertBrowse
TUS2-8	User	Browse browse = new Browse（）； BrowseMapper.updateBrowse（browse);	The time when the browsing record was updated in the database, return 1	Get 1	UpdateBrowse
TUS2-9	User	BrowseMapper.deleteBrowse(1，1);	Delete the browsing records whose userId is 1 and epId is 1 from the database, and return 1	Get 1	DeleteBrowse
TUS2-10	User	List browse = BrowseMapper.selectBrowseByUid（1);	Obtain a list of browsing records in the database according to userId	Get the list of Browse	SelectBrowseByUid
TUS2-11	User	BrowseMapper.existBrowse(1，1);	Browsing records with userId of 1 and epId of 1 are obtained in the database	Get the Browse object	ExistBrowse

2.2.2 Integration Testing

2.2.2.1 Summary of test results

After the unit test of the system, the integration test of the system includes the correctness inspection of the service layer interface of the system.

2.2.2.2 Detailed integration test results

Test case ID	Species	Condition	Desired result	Test Result	Test object ID
TUS1-1	Enterprise	EnterpriseForm enterpriseForm=new EnterpriseForm(); EnterpriseService.login(userForm);	Compare the EnterpriseForm information passed in from the front end with the Enterprise in the database for the password, correct and return to EnterpriseVO	Get EnterpriseVO	Login
TUS1-2	Enterprise	EnterpriseVO enterpriseVO= new EnterpriseVO(); EnterpriseService.createNewEnterPrise(enterpriseVO);	Register the enterprise, insert the EnterpriseVO information passed in from the front end into the database, and return 1	Get 1	CreateNewEnterPrise
TUS1-3	Enterprise	EnterpriseService.deleteEnterPrise(1);	The database deletes the Enterprise whose id is 1, and returns 1	Get 1	DeleteEnterPrise
TUS1-4	Enterprise	EnterpriseService.getEnterpriseById(7);	Get the Enterprise with id 7 in the database, and return EnterpriseVO	Get EnterpriseVO	GetEnterpriseById
TUS1-5	Enterprise	EnterpriseVO enterpriseVO= new EnterpriseVO(); EnterpriseService.updateEnterprise(enterpriseVO);	Update the Enterprise information in the database through the new EnterpriseVO passed in from the front end, and return 1	Get 1	UpdateEnterprise
TUS1-6	Enterprise	List enterpriseImgs = EnterpriseService.getEnterpriseImgList();	Get the url of the company registration image, and return List	Get List	GetEnterpriseImgList
TUS1-7	Enterprise	List enterpriseVOs= EnterpriseService.searchEpByKey(String key);	According to the keywords passed in by the front end, search for the matching Enterprise in the database and return List	Get List	SearchEpByKey
TUS1-8	Enterprise	EnterpriseService.updateTable(1);	Update all scores in the same type of enterprise table according to the incoming id, here is 1, update the scores of discrete large-scale enterprises, no return value	null	UpdateTable
TUS2-1	User	UserForm userForm= new UserForm(); UserService.login(userForm);	Compare the UserForm information passed in from the front end with the User in the database for the password, correct and return ResponseVO	Get ResponseVO or User information	Login
TUS2-2	User	UserVO userVO= new UserVO(); UserService.createUser(userVO);	Register an ordinary user, insert the UserVO information passed in from the front end into the database, and return ResponseVO	Get ResponseVO, that is, the information of successful registration	CreateUser
TUS2-3	User	UserVO userVO= new UserVO(); UserService.updateUser(userVO);	Update the User information in the database with the new UserVO passed in from the front end, and return 1	Get 1	UpdateUser

Test case ID	Species	Condition	Desired result	Test Result	Test object ID
TUS2-4	User	UserService.deleteUserById(2);	The database deletes User with id 2 and returns 1	Get 1	DeleteUserById
TUS2-5	User	UserService.getUserInfo(9);	Get the User information with id 9 from the database, and return UserVO	Get UserVO	GetUserInfo
TUS2-6	User	UserService.getUserNum();	Get the number of users	Get Integer	GetUserNum

2.2.3 function test

2.2.3.1 Summary of test results

The functional test of the system confirms that the observed behavior of the software complies with the software requirement specification, and verifies the correctness of the functional test from the perspective of the requirement use case.

2.2.3.2 Functional test detailed results

Test case ID	Species	Condition	Desired result	Test Result	Test object ID
TUS1-1	Enterprise	Double[] fin = Double[] fin = {4.0,0.52,2.0,2.5,0.5,2.0,0.05,0.5,1.0,1.0,0.3,0.1,0.5,166.0,2.0,3.0,55.0,1.0}; Double[] div = {95.0,3.0,1.0,-2.0,1.0,0.0,1.0,30.92,2.0,1.0,1.0,8.0,2.0,1.0,2.0,3.0,2.0}; ResponseVO rv = indexController.addEnterpriseTarget(1, div, fin);	Add indicators to the database and return ResponseVO	Get ResponseVO, that is, successfully add the indicator	addEnterpriseTarget
TUS1-2	Enterprise	Double[] fin = {0.0,0.51,2.0,2.5,0.5,2.0,0.05,0.5,1.0,1.0,0.3,0.1,0.5,166.0,2.0,3.0,55.0,1.0}; ResponseVO rv = indexController.updateEnterpriseFinanTarget(2, fin);	Update the economic indicator with id 2 in the database and return ResponseVO	Get ResponseVO, that is, successfully add the indicator	updateEnterpriseFinanTarget
TUS1-3	Enterprise	Double[] div = {50.0,4.0,2.0,-5.0,5.0,3.0,3.0,39.43,9.0,1.0,5.0,65.0,3.0,1.0,4.0,4.0,4.0}; ResponseVO rv = indexController.updateEnterpriseDiverTarget(2, div);	Update the two indicators in the database with id 2 and return ResponseVO	Get ResponseVO, that is, successfully add the indicator	updateEnterpriseDiverTarget
TUS1-4	Enterprise	ResponseVO responseVO = indexController.getEnterpriseTarget(1);	Get the index of the company with id 1 in the database, and return ResponseVO	Get index	getEnterpriseTarget
TUS2-1	User	UserVO userVO = new UserVO(); BeanUtils.copyProperties(user, userVO); UserForm userForm = new UserForm(); userForm.setEmail(user.getEmail()); userForm.setPassword(user.getPassword()); ResponseVO result = userController.login(userForm);	Compare the information in UserVO passed in from the front end with the User information in the database, and return ResponseVO	Get User information	Login
TUS2-2	User	UserVO userVO = new UserVO(); ResponseVO result = userController.register(userVO);	Insert the UserVO information from the front end into the database and return ResponseVO	Inserted successfully	register
TUS2-3	User	UserVO userVO = new UserVO(); ResponseVO result = userController.updateUser(userVO);	Update the UserVO information passed in from the front end to the database and return to ResponseVO	update completed	updateUser
TUS2-4	User	ResponseVO result = userController.deleteUser(2);	Delete User with id 2 from the database, and return ResponseVO	successfully deleted	deleteUser
TUS2-5	User	ResponseVO result = userController.getUserInfo(4);	Get User with id 4 from the database, and return ResponseVO	Acquired success	getUserInfo

3 other

· For detailed test case code, refer to the test file in the project file.